TEKNOLOGI MIKRO SERVICES

WEB SERVICE

Febri Damatraseta F, S.T, M.Kom

Material of Microservices

01
Introduction

Architecture of Monolith and Microservices

02
API Gateway

Authentification & Authorization

03
Spring Boot

Develop Microservices using JAVA

04
Development I

Build first project of Services Node

05
Development II

Busines transaction for Services Node

06
Review

Review Materi for UTS

Material of Microservices

07
API Manager

Set up API manager for services node

08
API Manager II

Set up API manager for services node

09
Presentation I

Team Project
Skema Services

10
Presentation II

Team Project
Progress

11
Presentation III

Team Project
Final

12
Review

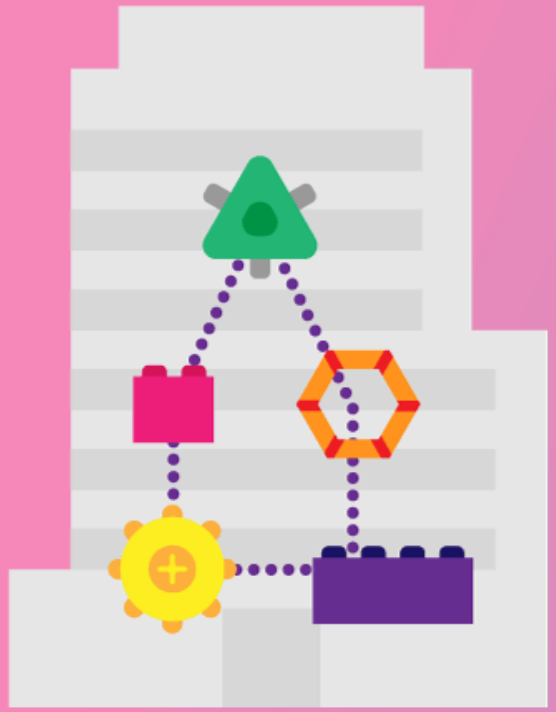
Review Materi untuk UAS

Kampus Merdeka
INDONESIA JAYA

INSTITUT BISNIS DAN INFORMATIKA KESATUAN
KESATUAN
BOGOR

01

Architecture of Monolith & Microservices



Architecture Monolith ?

Monolith merupakan salah satu teknologi yang saat ini masih digunakan.
Monolith adalah:

- Single Deployment Unit
- Dimana semua fitur dibuat dalam sebuah aplikasi besar

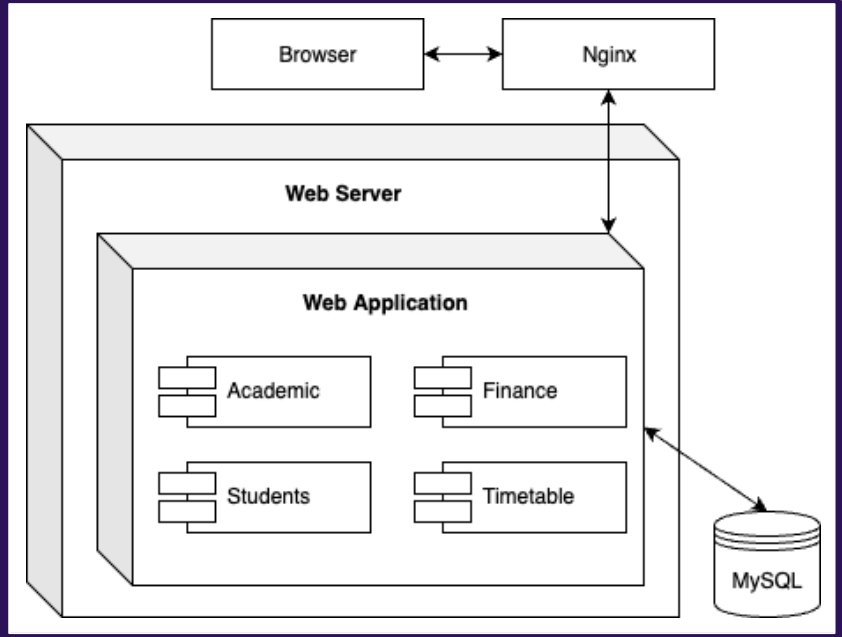
Biasanya aplikasi ini dibangun untuk kebutuhan fungsionalitas yang cukup besar. Namun teknologi ini mulai mengalami transisi ke Microservices.

Arsitektur Monolith

Seluruh code atau component menjadi satu rumah.

- Project application (Fontend & Backend)
- Database
- Library

Seluruh component diatas akan disimpan didalam satu server yang sama, contohnya menggunakan NGINX.



Kelebihan Monolith

Develop

Mudah dilakukan development

Deploy

Mudah ketika melakukan deployment

Test & Scale

Mudah dalam melakukan test and scale

Kekurangan Monolith

Develop

Menyusahkan developer baru

Scalling

Banyak developer maka banyak scalling.
Tidak bisa scaling bagian tertentu.

Technology

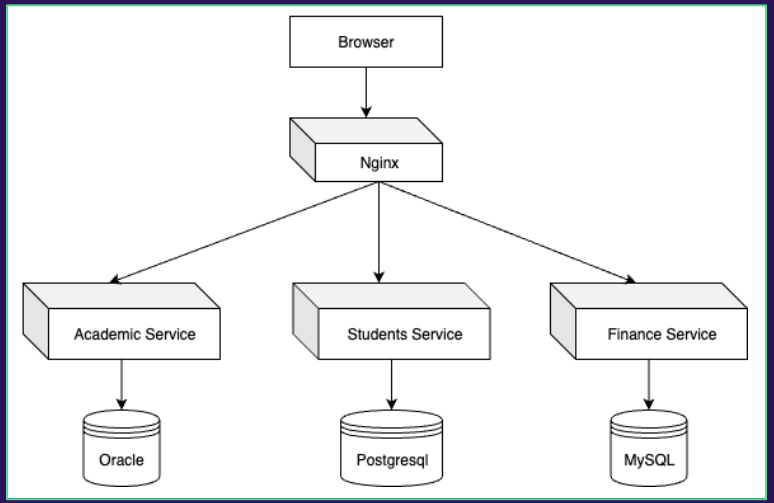
Terikat kontrak Panjang dengan Bahasa pemograman

Architecture of Microservices

Arsitektur ini menggunakan pendekatan SDLC yang didasari pada aplikasi.
Microservices:

- Aplikasi-aplikasi kecil yang saling bekerja sama.
- Fokus mengerjakan satu pekerjaan dengan baik
- Independent, dapat di deploy dan diubah tanpa tergantung dengan aplikasi lain
- Setiap komponen pada sistem dibuat dalam service
- Komunikasi antar service biasanya melalui network-call

Arsitektur Microservices



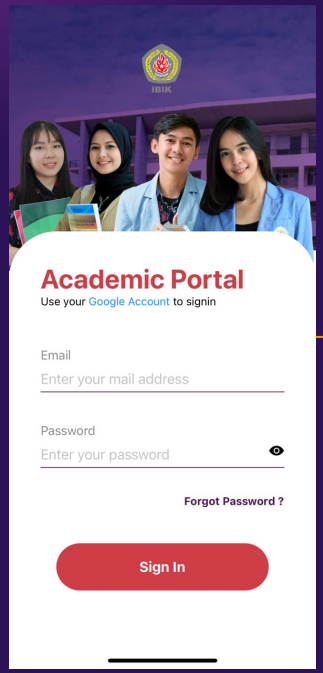
Kelebihan Microservices

- Mudah dimengerti, karena relative kecil ukuran service nya
- Lebih mudah di develop, di maintain, di test dan di deploy
- Lebih mudah bergonta-ganti teknologi
- Mudah di scale sesuai kebutuhan
- Bisa dikerjakan dalam tim-tim kecil

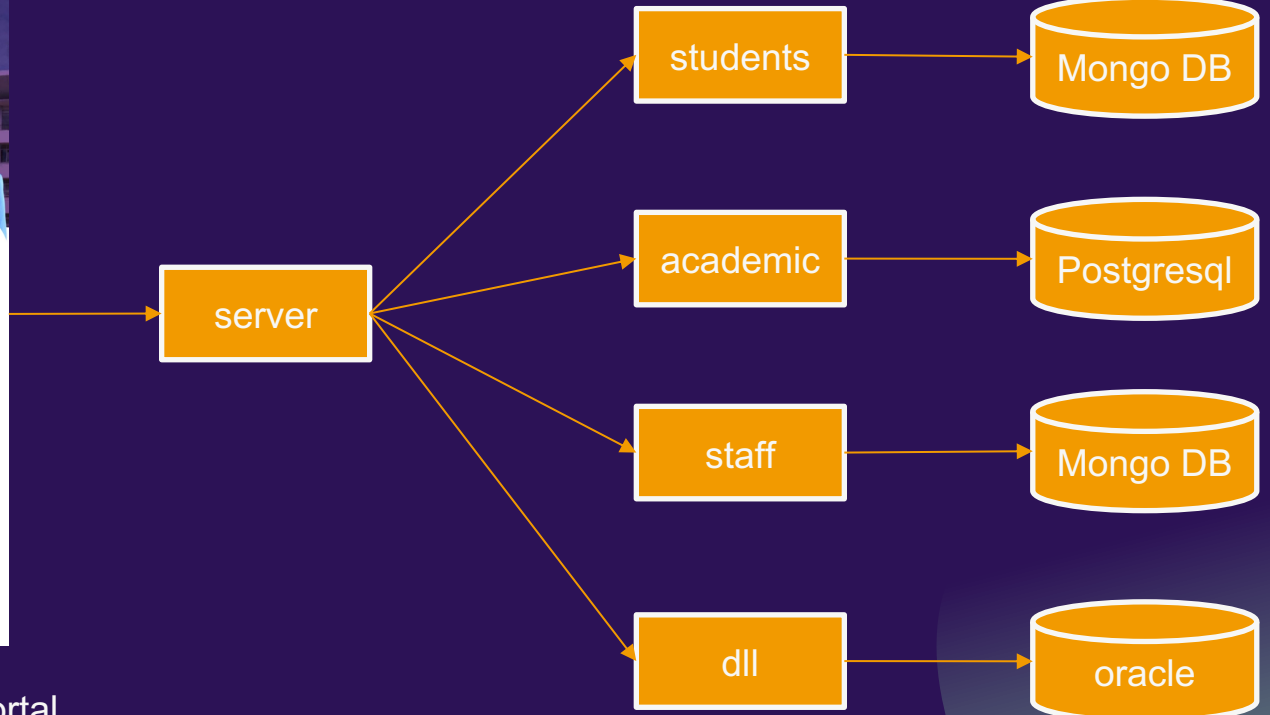
Kekurangan Microservices

- Distributed system
- Komunikasi antar service yang rawan error
- Testing interaksi antar service lebih sulit

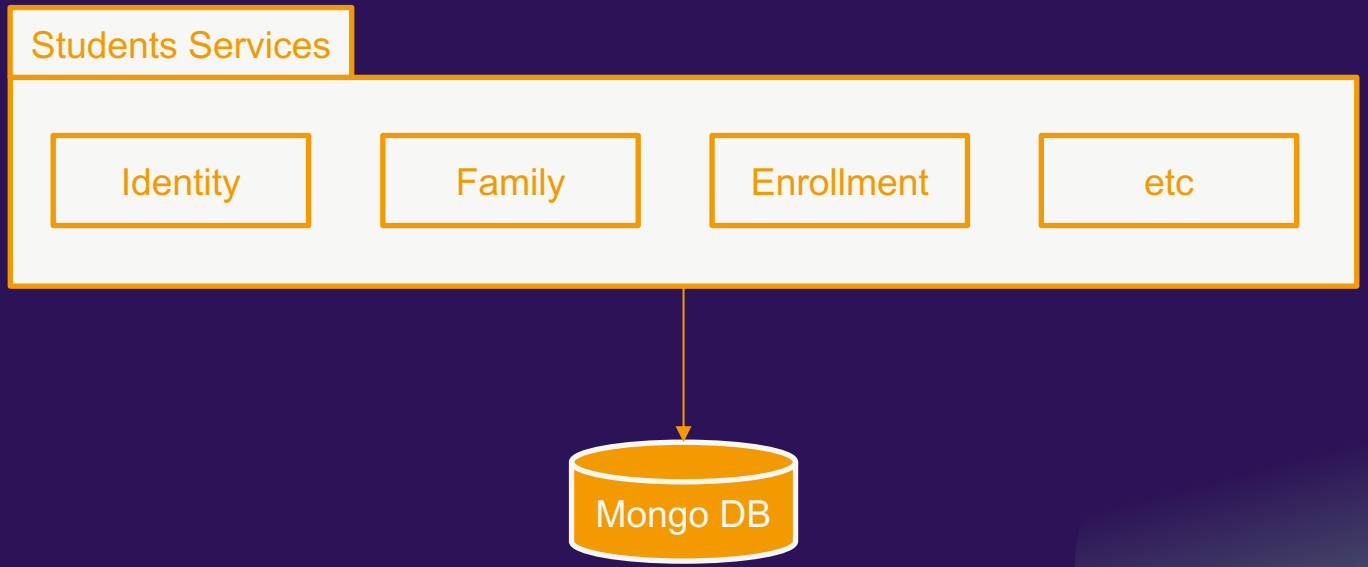
Contoh Pembagian Microservices



Aplikasi University Portal



Contoh Pembagian Microservices



Perbedaan

- Simplicity
- Consistency
- Easy to Refactor

- Partial Deployment
- Availability
- Multiple Platform
- Easy to Scale

02

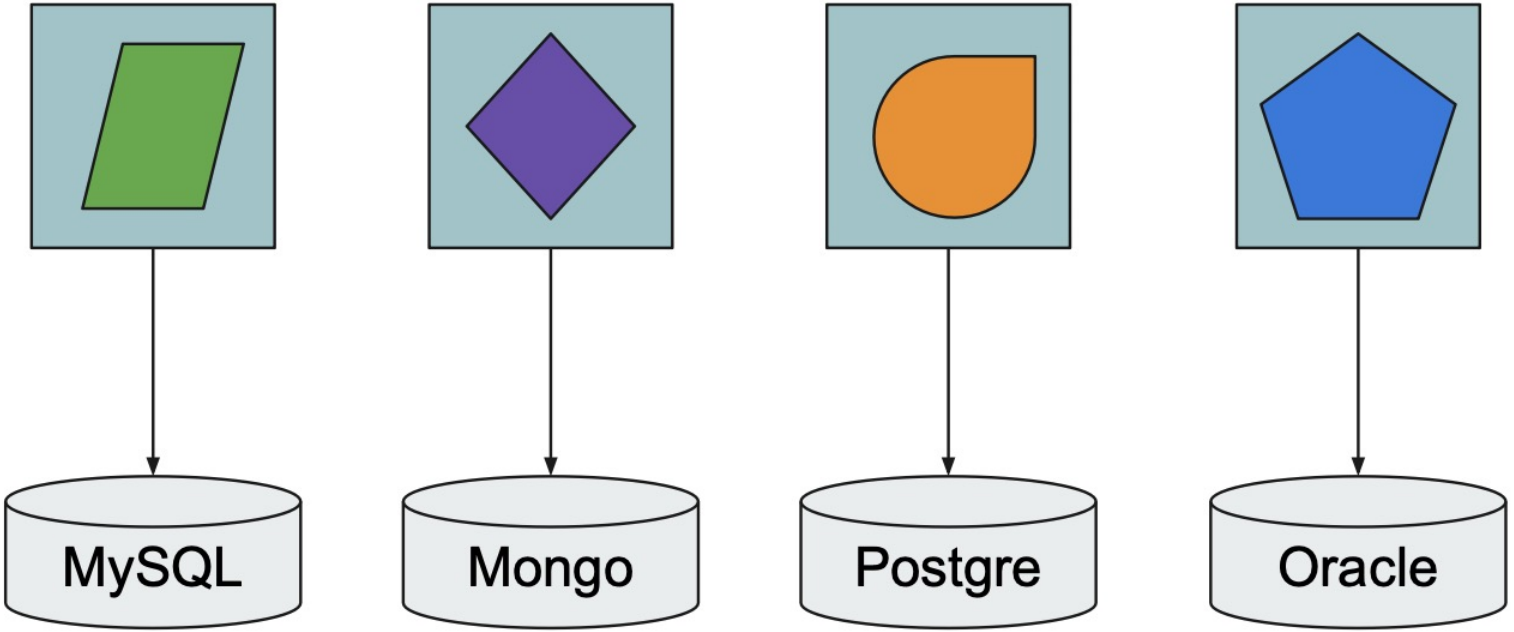
Database

Services & Shared

Pemilihan database per services

Database services

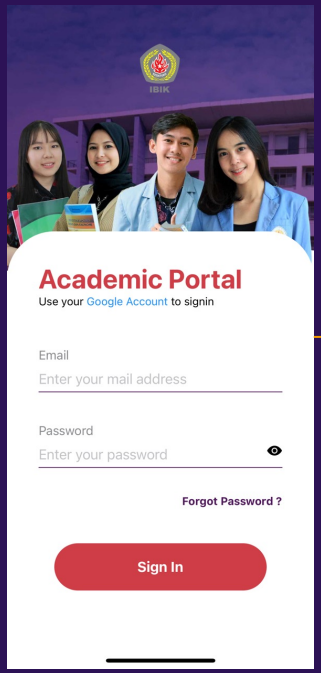
Decentralized Database



Why Decentralized Database ?

- Memastikan bahwa antar service tidak ketergantungan
- Tiap service bisa menggunakan aplikasi database sesuai dengan kebutuhan
- Service tidak perlu tahu kompleksitas internal database service lain

Database Services



Aplikasi University Portal

server

students

academic

staff

dll

Mongo DB

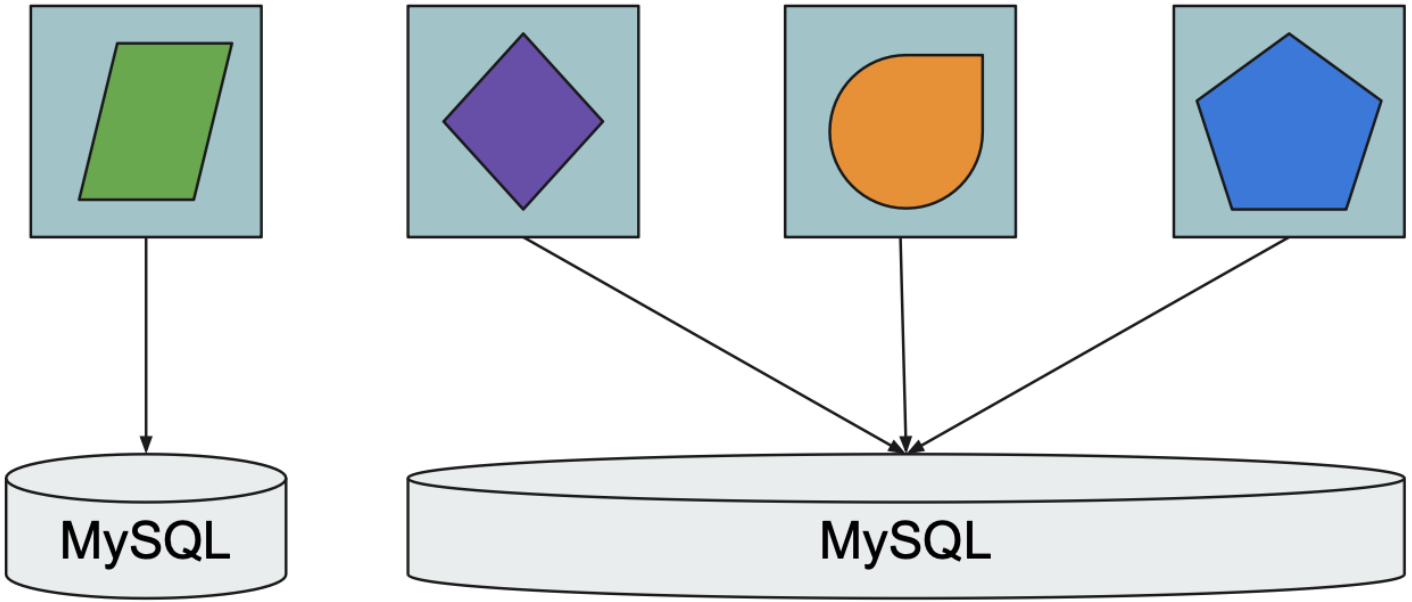
Postgresql

Mongo DB

oracle

Database shared

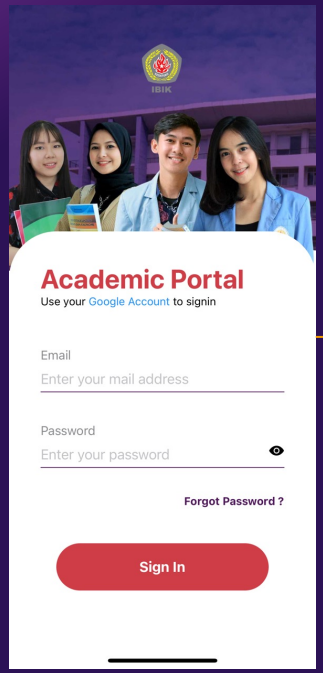
Database Shared

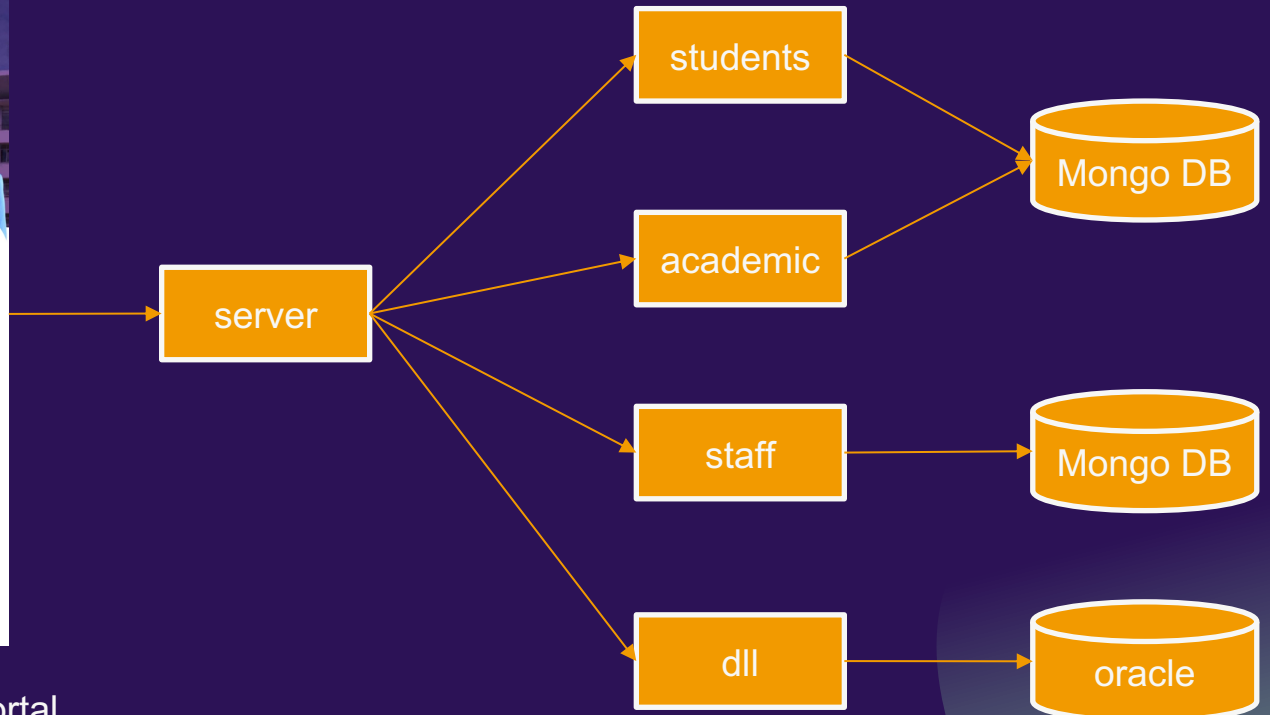


When use Database Shared ?

- Ketika melakukan transisi dari aplikasi Monolith ke Microservices
- Ketika bingung memecahkan data antar Service
- Ketika dikejar waktu, sehingga tidak ada waktu untuk bikin API

Database Shared





Aplikasi University Portal

03

Database

NO SQL

Not Only SQL

“NoSQL bisa mengelola database dengan skema yang fleksibel dan tidak membutuhkan query yang kompleks”

NoSQL Sample

Document Oriented Database

Output berupa JSON atau XML

Key and Value

Hanya terdapat dua buah kolum pada 1 table

Graph-based

menyimpan hubungan antar entitas dalam bentuk NODE

Column-based

menyimpan data dalam bentuk kolom. cocok untuk query SUM, COUNT, AVG, MIN,

04 Remote Procedure Invocation

Remote Procedure Call

“Dirancang untuk komputasi terdistribusi dan didasarkan pada semantik panggilan prosedur lokal

RPI / RPC

RPI / RPC Sample

Idealnya komunikasi dilakukan melalui RPI (Remote Procedure Invocation) atau RPC (Remote Procedure Call)

Tidak direkomendasikan komunikasi dilakukan via database

Komunikasi antar Service

Contoh RPI / RPC

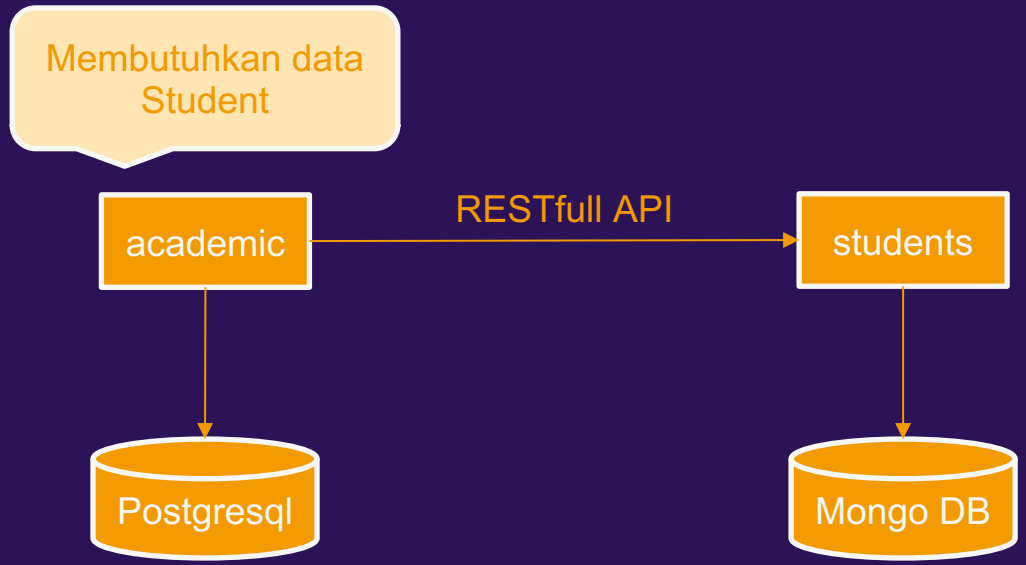
RESTful API (HTTP)
Application Program Interface

SOAP
Simple Object Access Protocol

gRPC
Google Remote Procedure Call

CORBA
Common Object Request Broker Architecture

RPI / RPC Sample



Sederhana dan Mudah

Biasanya digunakan untuk komunikasi Request - Reply

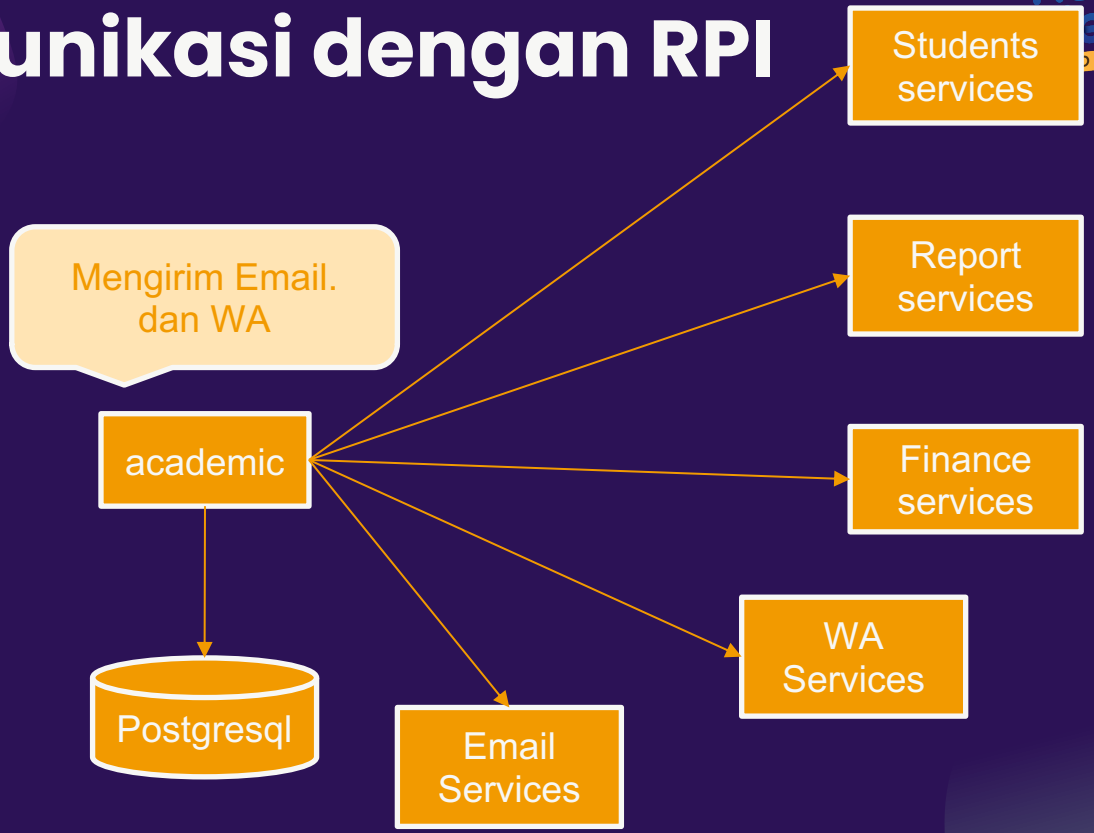
Biasanya digunakan untuk proses Sync (yang butuh menunggu jawaban)

Keuntungan

Messaging

Contoh kasus

Komunikasi dengan RPI



Masalah komunikasi di RPI

Proses eksekusi pengiriman Email, WA, SMS akan lambat

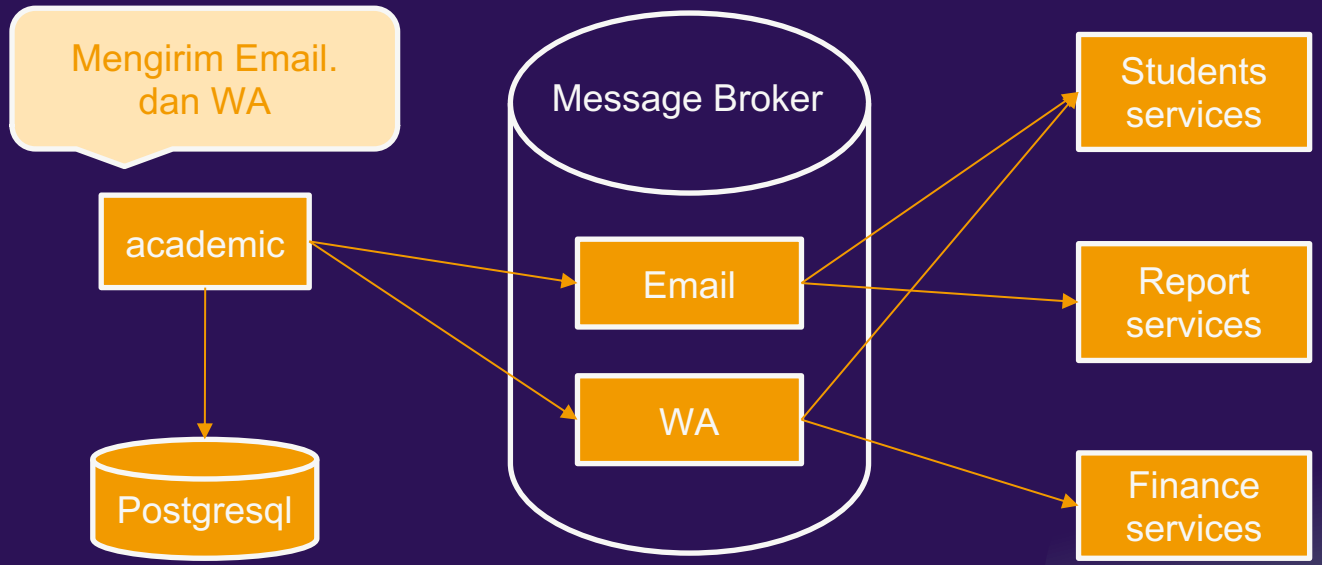
Mengirimkan data yang sama berkali-kali ke beberapa service

Membuat pararel process menjadi kompleks atau rumit

Komunikasi dengan Messaging

- Messaging biasanya digunakan untuk komunikasi Async
- Async artinya komunikasi dilakukan tanpa harus menunggu selesai di proses
- Dalam async, kadang tidak perlu peduli balasan dari service yang dituju
- Biasanya komunikasi Messaging membutuhkan Message Channel sebagai jembatan untuk mengirim dan menerima data
- Direkomendasikan menggunakan aplikasi Message Broker untuk melakukan management
- Message Channel

Komunikasi dengan Messaging



Proses lebih cepat karena tidak harus menunggu response

Service pengirim data tidak perlu peduli terhadap penerima data

Keuntungan

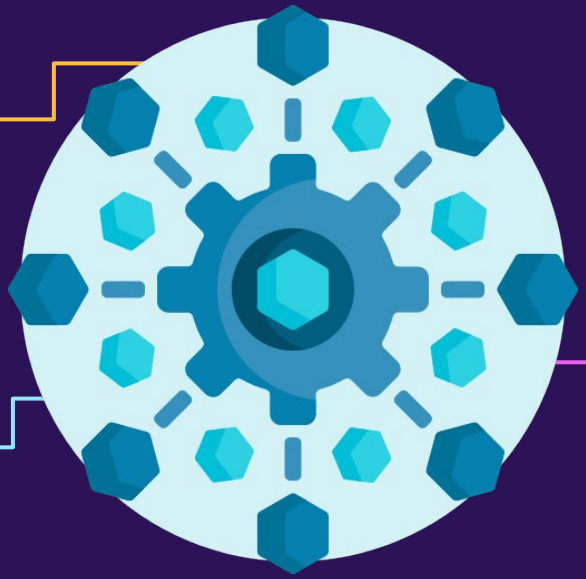
Type of Microservices

05

Type of Microservices

Stateless Microservice

Persistence Microservice



Aggregation Microservices

Stateless Microservice

- Biasanya tidak memiliki database
- Digunakan untuk melakukan tugas sederhana
- Biasa digunakan juga sebagai utility untuk microservice lain
- Tidak bergantung dengan microservice lain

Contoh service:

Email Services

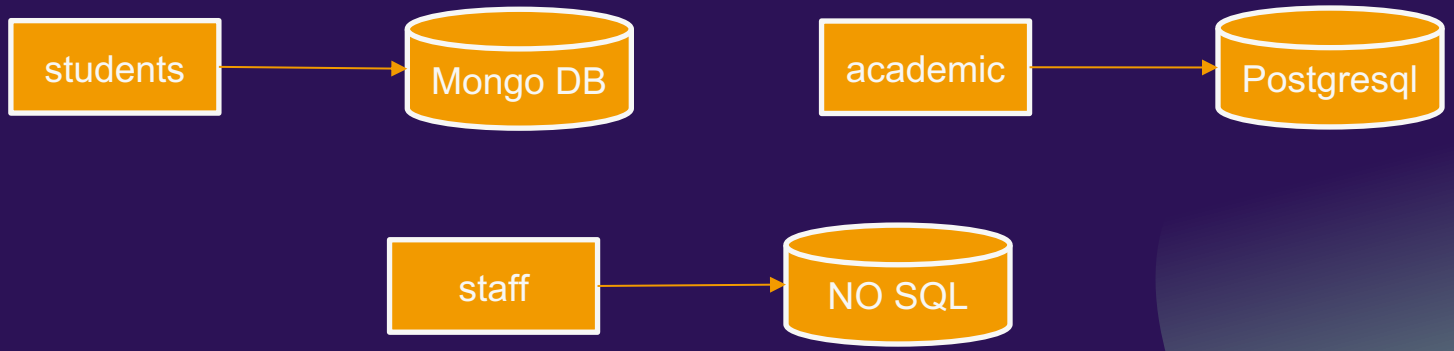
WA Services

SMS Services

Persistence Microservices

- Biasanya memiliki database
- Bisa juga disebut sebagai Master Data Microservice
- Biasa digunakan untuk mengolah data di database (CRUD)

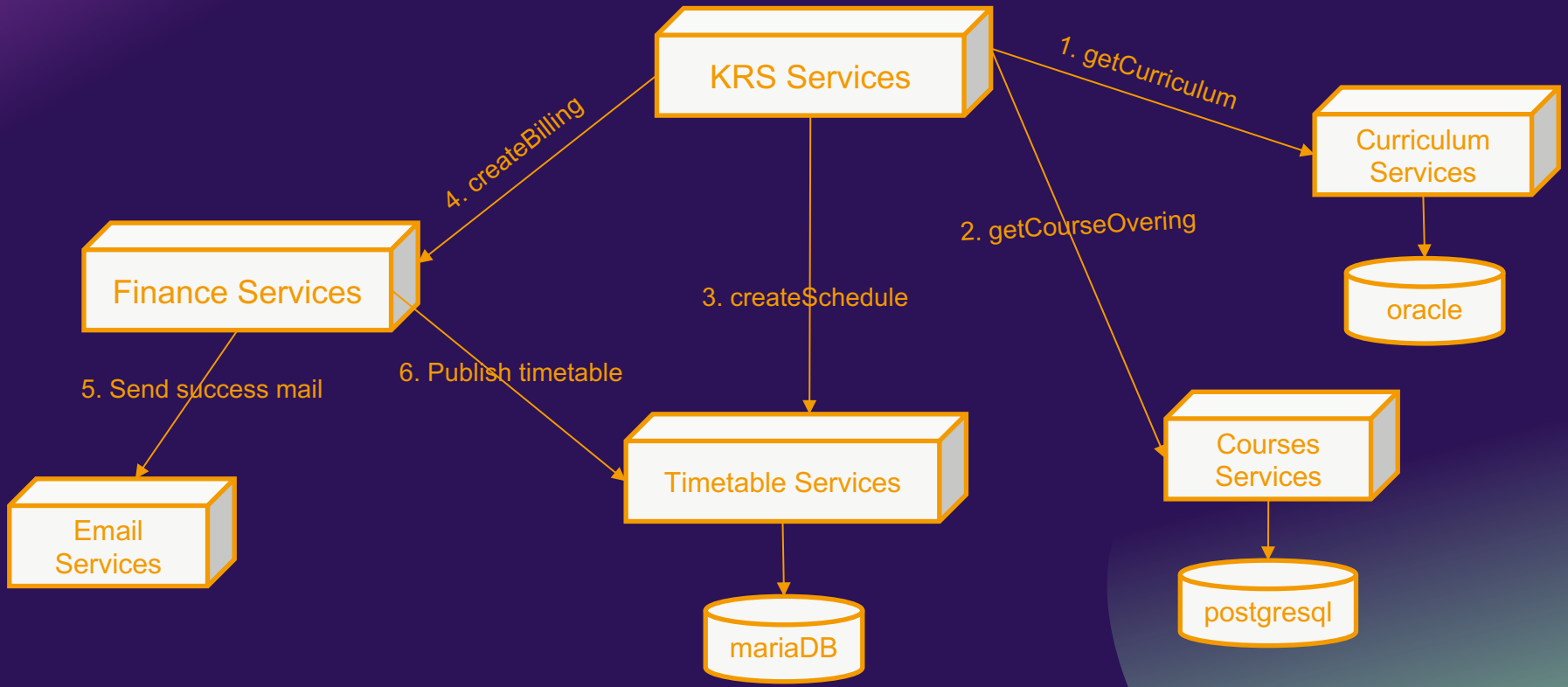
Contoh service:



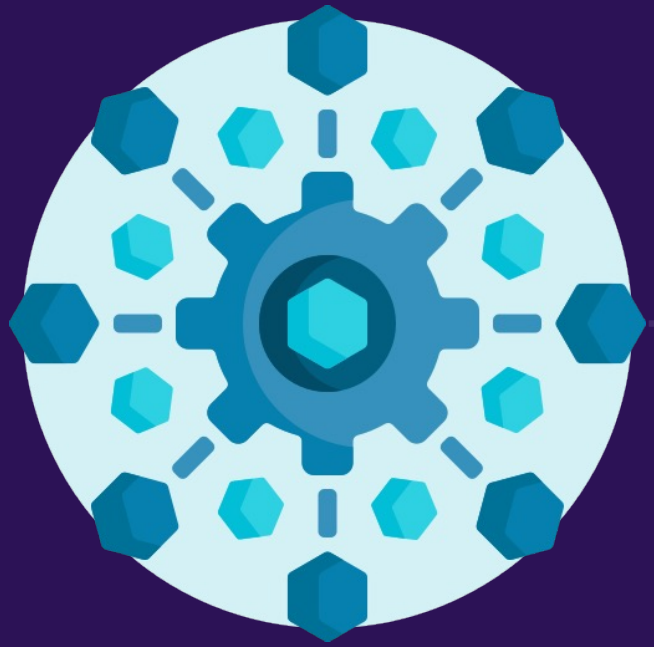
Aggregation Microservices

- Tergantung dengan microservice lain
- Biasa digunakan sebagai pusat business logic aplikasi
- Boleh memiliki database ataupun tidak
- Tidak bisa berdiri sendiri

Contoh Aggregation Microservices



Aggregation Microservices



Service Orchestration

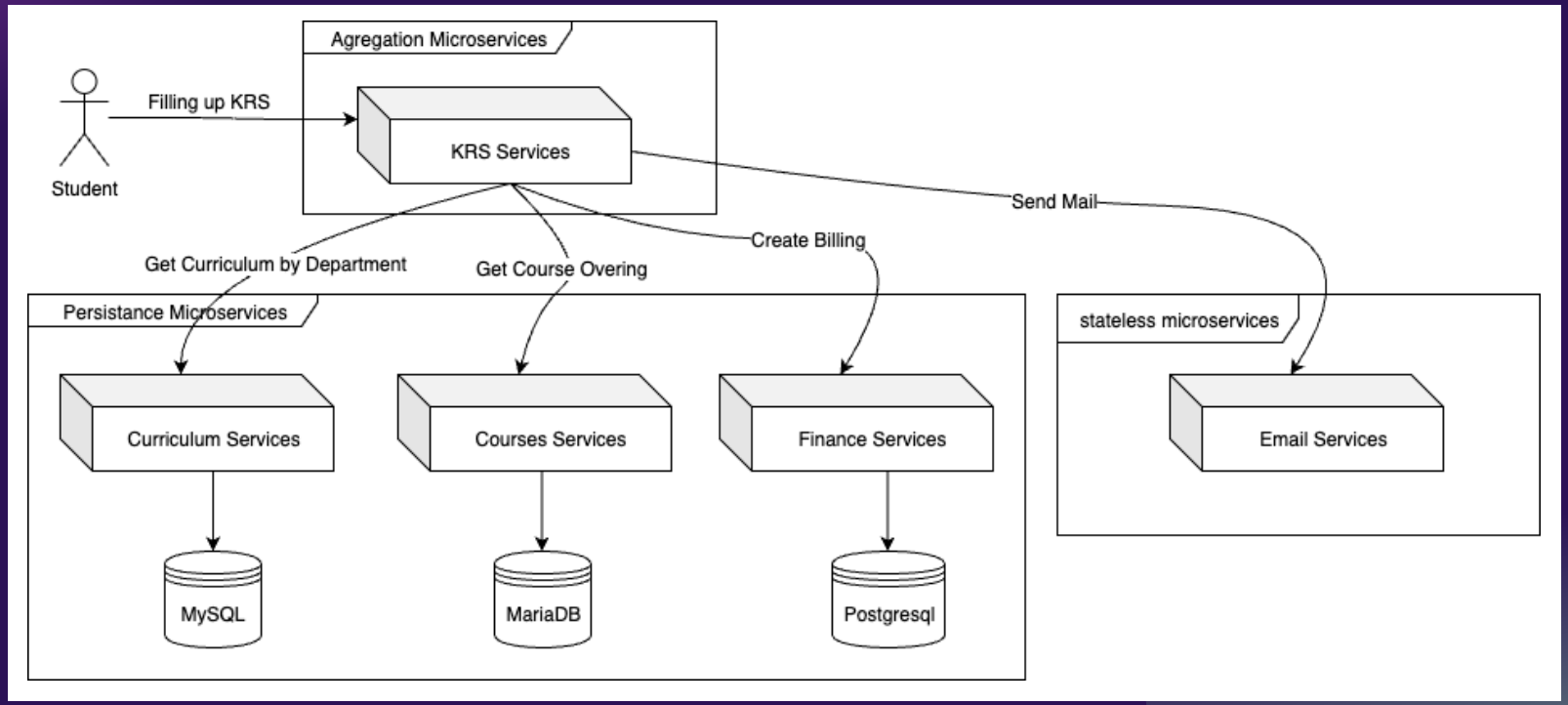
Service Choreography

Service Orchestration

- Cara Aggregation Microservices berkomunikasi dengan Microservices lain, jika menggunakan Remote Procedure Invocation, maka dinamakan ***Service Orchestration Pattern***

- Dalam ***Service Orchestration Pattern***, Aggregation Microservices bertugas untuk mengatur alur business logic sistem.

Service Orchestration



Keuntungan Service Orchestration

Mudah dibuat, karena kode business logic akan terpusat di Aggregation Microservices

Mudah dimengerti, karena kode business logic akan terpusat di Aggregation Microservices

Kekurangan Service Orchestration

Terlalu ketergantungan dengan Microservices lain

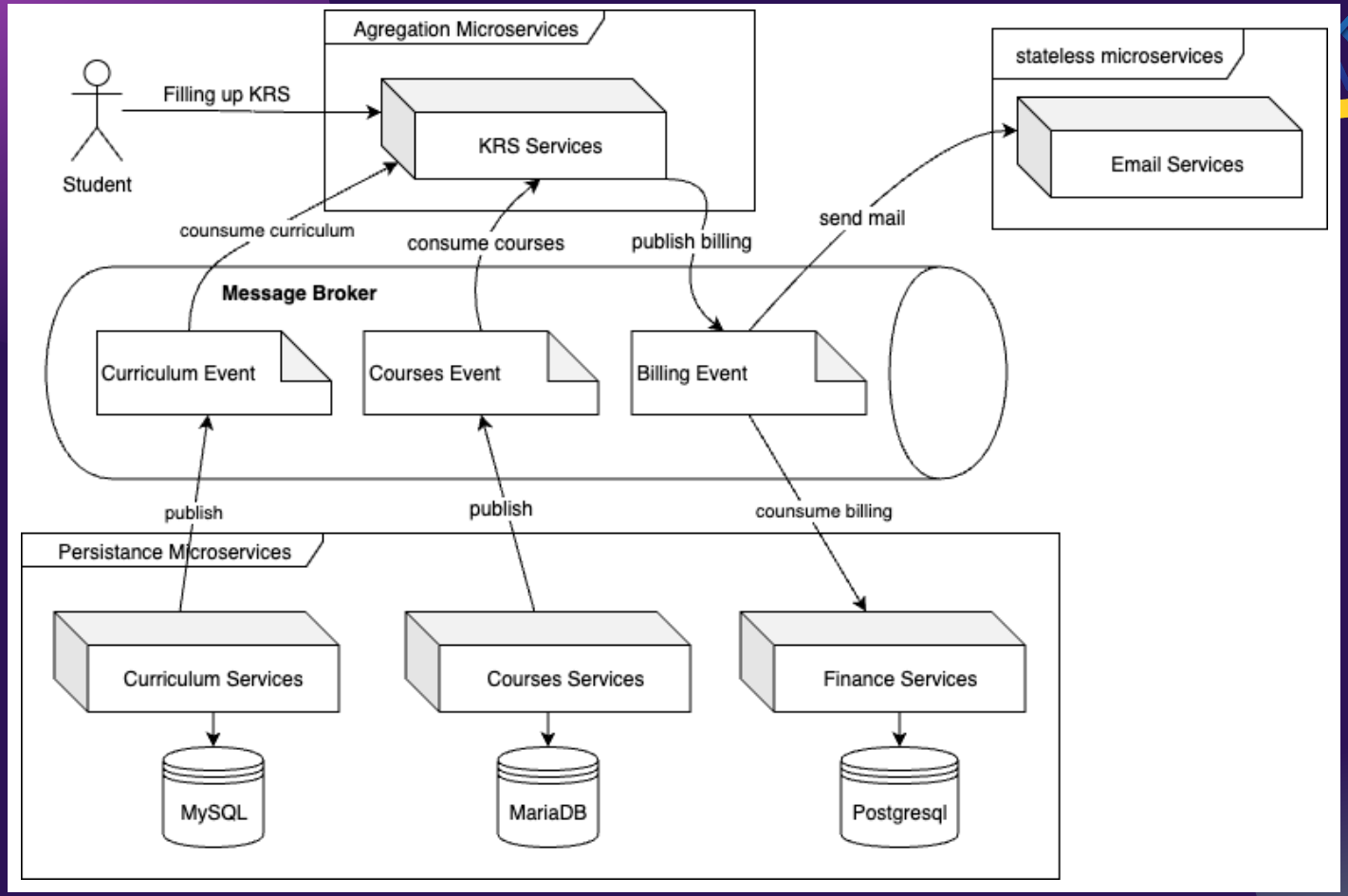
Akan lebih lambat karena harus terkoneksi dengan Microservices lain

Akan lebih mudah error jika di Microservices lain terdapat masalah

Jika perlu Microservices baru, perlu dilakukan perubahan di Aggregation Microservices

Service Choreography

- Dalam Service Choreography, komunikasi Aggregation Service dengan Microservices lainnya menggunakan ***Messaging***.
- Dalam Service Orchestration, Aggregation Microservice adalah service yang sangat kompleks dan mengerti semua alur business logic,
- Service Choreography, semua Microservices dituntut untuk menjadi pintar, tidak hanya diperintah oleh Aggregation Microservices.



Kampus Merdeka INDONESIA JAYA

INSTITUT BISNIS DAN INFORMATIKA KESATUAN BOGOR

Service Choreography

Keuntungan Service Choreography

Aggregation Microservices tidak tergantung dengan Microservices lainnya

Aggregation Microservice akan lebih cepat, karena tidak perlu berkomunikasi dengan Microservices lainnya

Jika ada Microservice baru, Aggregation Microservice tidak perlu melakukan perubahan lagi

Kekurangan Service Choreography

Lebih sulit di-debug ketika terjadi masalah

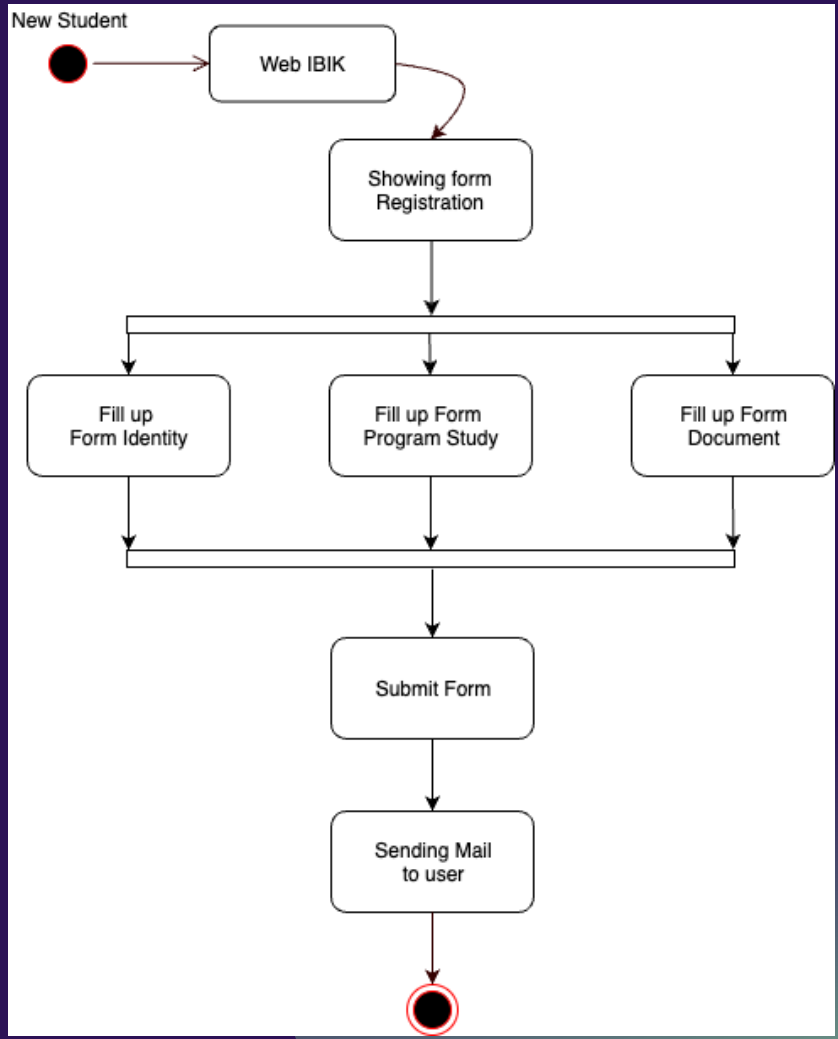
Business logic akan terdistribusi di semua Microservices, sehingga sulit untuk dimengerti secara keseluruhan

Assignment

Tugas Individu

Assignment

- Dari rancangan Activity Diagram disamping. Buatlah skema untuk Arsitektur Microservices. Tentukan setiap service berdasarkan tipe microservices.
- Kumpulkan dalam bentuk document PDF yang berisi rancangan skema Aristektur Microservices.
- Kirimkan ke febrid@ibik.ac.id dengan subject TMS-01-TI-20-PA-NPM paling lambat Jumat 16 Sep 2022 pukul 23.00



Thanks!

Does anyone have any questions?

febrid@ibik.ac.id
+62 813 9889 4710